# ZPR - gra przeglądarkowa z użyciem WebAssembly

Dokumentacja wstępna

Krzysztof Pałucki, Alicja Kubiszyn

## Opis projektu

Celem projektu jest implementacja aplikacji pozwalającej na dwuosobową grę w szachy w oparciu o architekturę klient-serwer, klient będzie wykorzystywał jedynie przeglądarkę www. Aplikacja będzie wykorzystywać WebAssembly w celu komunikacji między logiką gry a interfejsem graficznym.

## Lista funkcjonalności i wymagań

**Zadbanie o reguły rozgrywki**
Aplikacja powinna zadbać o poprawną implementację reguł szachów, w tym takich jak: bicie w przelocie, promocja, roszady, itd. Logika gry powinna być świadoma kiedy jest szach, na jakie pola może się udać dana figura, kiedy nastąpił mat.

**Gra dwuosobowa z wykorzystaniem internetu**
Aplikacja powinna pozwolić na łączenie się z drugą osobą grającą na innej maszynie za pomocą połączenia internetowego. Służyć ma temu serwer, który zapewni łączność oraz pomoże w uwierzytelnieniu rozgrywki.

**Gra dwuosobowa na zasadzie “hotseat”**
Aplikacja powinna umożliwić grę w szachy pomiędzy dwoma graczami, którzy dzielą jedno urządzenie i korzystają z funkcji “hotseat” w trybie lokalnym.

**Uwierzytelnianie**
Aplikacja powinna zapewnić sposób na uwierzytelnienie rozgrywki tak, aby obydwoje gracze byli w stanie przeprowadzić ze sobą mecz. Będzie to spełnione przez generowany w trakcie tworzenia meczu kod gry, za pomocą którego druga osoba może dołączyć do odpowiedniej partii.

**Interfejs użytkownika**
Interfejs powinien zapewnić następującą funkcjonalność:

- menu główne umożliwiające wybór rodzaju gry (opcja nr 1/ opcja nr 2), w przypadku opcji nr 1 wygenerowanie lub wpisanie kodu rozgrywki
- pole umożliwiające wpisanie imion graczy
- poprawne wyświetlanie szachownicy oraz jej stanu (w tym m.in. wyświetlanie szachownicy z perspektywy białych/czarnych w zależności od gracza)
- przycisk do poddania się
- przycisk do remisu
- opcjonalnie: zegar, który odlicza czas do końca limitu gracza

**Kompatybilność**
Aplikacja powinna działać na różnych systemach (Windows, Linux) i przeglądarkach (Chrome, Firefox, etc.) po stronie klienta.

## Narzędzia i środowisko deweloperskie

Utworzenie aplikacji będzie podzielone na dwie części: backend (silnik gry oraz serwer) oparty na WebAssembly i napisany w Ruście oraz frontend (interfejs użytkownika) w TypeScripcie. Komunikacja z użytkownikami będzie opierała się o mechanizm WebSocketów. Kod będzie przechowywany na repozytorium wydziałowym na GitLabie. Środowisko do programowania w Ruście będzie oparte na oficjalnym systemie narzędzi (rustup, cargo, itd.), podczas gdy część odpowiedzialna za frontend - na środowisku npm. Za generowanie dokumentacji po stronie frontendu odpowiedzialna będzie biblioteka JSDoc za testowanie - Vitest, a za formatowanie - Prettier (po stronie backendu będą to odpowiednio cargo (testowanie oraz dokumentacja) oraz rustfmt).